Recognition Tasks are Imitation Games

Richard Zanibbi[1], Dorothea Blostein[2], and James R. Cordy[2]

[1] Centre for Pattern Recognition and Machine Intelligence,
1455 de Maisonneuve Blvd. West, Suite GM 606, Montreal, Canada, H3G 1M8
zanibbi@cenparmi.concordia.ca
[2] School of Computing, Queen's University, Kingston, ON, Canada, K7L 3N6
{blostein,cordy}@cs.queensu.ca

Abstract. There is need for more formal specification of recognition tasks. Currently, it is common to use labeled training samples to illustrate the task to be performed. The mathematical theory of games may provide more formal and complete definitions for recognition tasks. We present an imitation game that describes a wide variety of recognition tasks, including the classification of isolated patterns and structural analysis. In each round of the game, a set of 'players' try to match the interpretation of an input produced by a set of 'experts.' The 'playing field' on which experts and players operate is a set of interpretations generated from legal sequences of 'moves' for a round. The expert and player moves transform interpretations, and select interpretations for output. The distance between interpretations in the playing field is defined by a distance metric for interpretations, and the game outcome by a ranking function on distance values observed for players' interpretations. We demonstrate how this imitation game may be used to define and compare recognition tasks, and clarify the evaluation of proposed solutions.

1 Introduction

Many recognition tasks have strong similarities to the children's game 'pin the tail on the donkey.' In that game, players are shown a picture of a donkey without a tail. Players take turns being blindfolded, turned around several times, and then trying to place a pin with an attached tail at the proper location on the donkey. At the end of the game, the adult running the game awards prizes to the children based on the closeness of their pins to the 'proper' location. At first, 'pin the tail on the donkey' might seem like a strange analogy for recognition tasks, but consider the following similarities.

1. **The basic task is to choose points in space.** In the game, a pin is used to pick a physical location within a room. In recognition, interpretations of an input or object are selected from a space of possible interpretations. The relative distances ('closeness') of interpretations within the space are determined by a distance metric (e.g. classification risk or edit distance).
2. **Goal points are determined by an *expert opinion*.** In the game, the adult chooses the optimal tail location(s) using his or her understanding

of donkey anatomy. For recognition tasks, one or more experts use their understanding of the problem domain to select goal interpretations.

3. **From an initial point in space, the goal point(s) must be *guessed.*** [1] In each turn of the game, a blindfolded and disoriented player must make a sequence of guesses as to how to move and eventually place their tail from their starting position. Similarly, despite ambiguities in an input's content or introduced by noise, a recognition protocol must make a sequence of guesses about which interpretations should be considered and/or selected as goal interpretations, starting from some initial interpretation (e.g. 'reject').
4. **Guesses are ranked using their *distance* from the expert opinion.** In the game, tail locations are ranked using the distance from pins to the location(s) chosen by the adult. For recognition tasks, recognition protocols are ranked using a function of the distances from guessed to goal interpretations within the interpretation space (e.g. the minimum, mean, or median of interpretation space distances observed for a test sample).

We propose that like the players in 'pin the tail on the donkey,' recognition systems evaluated against expert opinions are engaged in an *imitation game* where players producing responses that are *closest* to that of an expert opinion are deemed most successful. The outcome of such a game depends directly upon expert opinion(s), how 'closest' is defined, and the interpretation spaces used by experts and players, which may not coincide. As an extreme example, if the donkey picture in 'pin the tail' is placed too high for a child to reach the goal tail location(s), the child might not consider these locations (i.e. they do not exist in the child's interpretation space). Similarly, the use of different domain models by experts and recognition algorithms may produce 'holes' in the algorithms' interpretation spaces, which may prevent goal interpretations from being considered.

Using an imitation game to define recognition tasks places evaluation within problem definitions, as opposed to treating evaluation as a validation of proposed solutions for more abstract problems. This is similar to how Turing avoided directly considering whether machines "think" by instead considering outcomes of his own famous imitation game [3]. As an example, consider classifying images of handwritten digits (0..9) when the cost (risk) of classification error is fixed, as opposed to when it is not (i.e. solutions try to 'recognize digits,' in the absence of an explicit evaluation scheme). In the second scenario, an evaluation mechanism may be chosen *after* solutions for recognizing digits have been defined. This compares solutions unfairly, particularly if they are designed assuming different evaluation schemes. Evaluation in the context of an imitation game is more meaningful, because the assumptions ('rules') under which solutions ('players') operate are explicit and uniform. Imagine telling children after placing two tails each in 'pin the tail on the donkey' that they will be ranked by *mean* rather than *minimum* pin distance.

[1] Interesting discussions pertaining to guessing in pattern recognition have recently been provided by Kanatani [1] and Oommen and Rueda [2].

Games and game theory [4, 5] have of course been used previously in the pattern recognition literature. For example, game-theoretic models have been used for combining modules in vision systems [6, 7] and for modeling sequential prediction problems [8]. Based on earlier work for monitoring aircraft engines [9], Pau has modeled Bayesian classification using two-player games, with a classifier deciding the *a posteriori* probabilities of input patterns, and a teacher providing the *a priori* class probabilities to the classifier [10]. A zero-sum game between the teacher and classifier is used for the worst-case, when the teacher tries to maximize the number of errors made by the classifier. A bi-matrix game in which teacher and classifier may cooperate is also examined. Optimal strategies for player and teacher in each game (equilibrium solutions) are presented.

We are taking a different tack here, as we present a class of games for defining and comparing various recognition problems, including the classification of isolated patterns and structural analysis. The purpose of our imitation game is to compare recognition strategies rather than optimize a single one, as in Pau's game. For this discussion we assume that all our 'players' produce a sequence of decisions, leaving issues pertinent to parallelism for the future. We use examples from 'pin the tail on the donkey,' digit recognition, and table cell detection for illustration.

2 Rules of the Game: Interpretation Models

Let's consider a simple mathematical model for the rules of 'pin the tail on the donkey,' ignoring the vertical position of pins. We will model the position of a child and their tail-pin as a line segment in R^2, with the child at one point, and the pin at the other. The distances between pins will be defined by their Euclidean distance in R^2. During a turn, a child may do the following actions: move forward, turn varying amounts clockwise or counter-clockwise, and push their tail-pin forward to fix it in a wall. We model these actions as transforming the location of the line segment; walking forward translates the segment, turning rotates the 'pin' point around the 'child' point, and a successful push of the pin into the wall fixes the pin location, and ends the turn. If a child pushes their pin into empty space, the line segment remains unchanged, and the turn continues.

We now have rules for the game, defining the legal space of guesses (*interpretation space*) from the sequences of moves that a child may perform during their turn(s). If we include failed pin pushes (pushes into air), the locations a child may attempt to place the pin includes all the space in R^2 on and between the walls of the room. This provides us with a *generative* model of the interpretation space, in which a player's turn is described by a sequence of model operations (an *operation sequence*), and the interpretation(s) selected as a result.

In our imitation game for recognition tasks, we will define an *interpretation model* as a 7-tuple m:

$$m = (D^m, i_0^m, T^m, \Gamma^m, apply^m, \delta^m, time_{max}^m). \tag{1}$$

D^m is a set of problem domain inputs, the set of elements for which interpretations are constructed by the model. For a model m, the interpretation for all

inputs $d \notin D^m$ is defined as the initial interpretation, i_0^m. For a classification model, i_0^m would be 'reject', and for a structural recognition model, i_0^m might be the empty set.

T^m is a set of *model operations* that transform interpretations, the set of possible moves for the game. In 'pin the tail on the donkey,' these were the actions to move and push the tail-pin. Generally speaking, operations in T^m create, delete, classify, segment, and relate entities in interpretations [11]. T^m must include an *accept* operation which marks interpretations for output (as a final 'guess' in the game), and may also include a *reject* transform for reversing an *accept* operation. Model operations in T^m may alter *sets* of interpretations, such as for representing the combination of interpretations by a classifier ensemble, or for generating alternatives. As a simple example, T^m for an interpretation space of digit strings might contain *accept*, *reject*, and a set of string edit operations (e.g. replace a digit, transpose a digit, insert a digit, delete a digit).

Γ^m defines legal sequences of moves (the model operations, T^m) similar to a string grammar with start symbol i_0^m and terminals T^m. Restrictions on the maximum number of moves in a turn may be enforced by defining Γ^m such that legal transformation sequences have a finite maximum length k. All turns begin with the initial interpretation (i_0^m) as the default interpretation. The legal interpretation sequences are denoted L^m (the *language of model m*), and the interpretation space I^m is obtained using the function $apply^m$.

$$L^m = \Gamma^m(i_0^m, T^m) \tag{2}$$

$$I^m = i_0^m \cup \bigcup_{s \in L^m} apply^m(s, i_0^m) \tag{3}$$

$$apply^m(s \notin L^m, i_0^m) = i_0^m \tag{4}$$

The function $apply^m$ constructs interpretations by applying an operation sequence to the initial interpretation (i_0^m); illegal sequences are mapped to i_0^m.

We define one property of operation sequences, the *interpretation history* (h^m). The interpretation history of an operation sequence $s \in L^m$ is the set of unique interpretations generated over the course of applying the sequence.

$$h^m(s = \{t_1 \in T^m, .., t_{|s|} \in T^m\}) = \bigcup_{i=1}^{|s|} apply^m(t_1..t_i) \tag{5}$$

An interpretation history contains all unique interpretations constructed during recognition. This is the set of pin locations for a turn in 'pin the tail on the donkey.'

The distance function δ^m defines a distance between interpretations (this was Euclidean distance in R^2 for our simple 'pin the tail' model). Consider our example of the digit string interpretation space employing string edit operations in T^m. δ^m could be defined as a 'string edit distance,' the minimum number of operations needed to transform one digit string to another. Alternatively, if we use the numerical difference of the numbers represented by the strings for δ^m, the operation sequences and distances are less directly related.

The final component of an interpretation model m is $time^m_{max}$, the maximum time in which an operation sequence may be generated. Unbounded recognition time may be represented using ∞. For both 'pin the tail on the donkey' and real-world recognition problems, $time^m_{max}$ is finite, and relatively small.

3 Playing Recognition Games

We now define an imitation game for recognition tasks, in which a sample of a problem domain is taken, experts define goal interpretations for the sample elements, and players try to guess the expert interpretations. Experts and players use two models, differing only in the time for choosing interpretations. This allows for differences between human 'experts' (e.g. using a GUI to create interpretations) and algorithms under real-time constraints.

For a recognition game, the *recognition game parameters* (g) are an 8-tuple:

$$g = (m^g, time^g_e, time^g_p, \mu^g, n, \phi^g, E^g, P^g). \tag{6}$$

m^g is an interpretation model as described in the previous section. $time^g_e$ is the maximum time allowed for creating expert interpretation(s), and $time^g_p$ is the maximum duration of a player's turn. μ^g is a sampling function returning a list of q elements from a set (e.g. $\mu^g(D^{m^g}, q) = (d_1 \in D^{m^g}, ..., d_q \in D^{m^g})$); n is the sample size used in the game. ϕ^g is a ranking function, defining an ordering for sets of values (e.g. ranking by minimum distance, as in 'pin the tail on the donkey').

Players are represented as functions called *recognition strategies* ($P^g = \{\alpha_1, .., \alpha_p\}$) returning legal operation sequences from the language of m^g (for $i \in \{1..p\}$, $\alpha_i(d \in D^{m^g}, m^g, time^g_p) = s \in L^{m^g}$). The *expert protocol* defining goal interpretations is defined as a pair $E^g = (A^e, \beta)$, where $A^e = \{\alpha^e_1, .., \alpha^e_j\}$ is another set of recognition strategies using time restriction $time^g_e$, and β is a function combining the expert opinions (operation sequences) to produce goal interpretation(s) for an input $d \in D^{m^g}$ $(\beta(\alpha^e_1(d), ..., \alpha^e_j(d)) = \{i_1, ..., i_m\})$.

Given recognition game parameters g, a *recognition game* is an imitation game that proceeds as follows:

1. The game input set D^g is defined by

$$D^g = \mu(D^{m^g}, n) = \{d_1..d_n\} \subseteq D^{m^g}. \tag{7}$$

2. The goal interpretation set $(I^e_{d_k})$ for each input $d_k \in D^g, k = 1..n$ is defined using the expert protocol E^g
3. A series of 'rounds' is played, one for each $d_k, k = 1..n$. In each round:
   (a) Each player in P^g is given d_k and produces an operation sequence $(\{s^{\alpha_1}_{d_k} .. s^{\alpha_p}_{d_k}\})$, from which the guessed interpretation sets $(\{I^{\alpha_1}_{d_k}..I^{\alpha_p}_{d_k}\})$ are obtained using $apply^{m^g}$ (e.g. $I^{\alpha_1}_{d_k} = apply^{m^g}(s^{\alpha_1}_{d_k})$)

(b) Each player is scored using the distance function of the game interpretation model (δ^{m^g}), to produce a set of distances $\Delta_{d_k}^{\alpha_x}$ between each guessed interpretation in $I_{d_k}^{\alpha_x}$ and each goal interpretation in $I_{d_k}^{e}$:

$$\Delta_{d_k}^{\alpha_x} = \bigcup_{i^\alpha \in I_{d_k}^{\alpha x},\ i^e \in I_{d_k}^{e}} \delta^{m^g}(i^\alpha, i^e), \forall x = 1..p, \forall k = 1..n \tag{8}$$

4. The player ranking is determined by applying the ranking function ϕ^g to the scores from each round (e.g. $\phi^g(\{\Delta_{d_1}^{\alpha_1}..\Delta_{d_n}^{\alpha_p}\}) = (\alpha_3, \{\alpha_p, \alpha_1\}, ...)$, where α_3 wins, and α_p and α_1 are tied for second place)

4 Decision Making in Recognition Strategies

In our game, the experts and players are modeled by functions called *recognition strategies*, which return operation sequences for a problem domain input. Operation sequences represent decisions made within a series of *decision spaces* containing the alternatives for each decision; applying a model operation implies that some inference has been made regarding the appropriate model instance(s) for an input [11]. Sequential decision making can be represented by a decision tree, flow chart, or similar representation [4, 5]. We will consider properties of a decision tree representation for decision making in our recognition games.

Consider Figure 1, which presents a single turn for 'pin the tail on the donkey.' Shown are the sequence of decisions made ($s = (forward,\ turn\ 5\ deg.\ CC,\ push)$), and the set of alternatives considered. At each point in the tree, some alternatives will produce identical pin locations. For example, deciding to turn 30° clockwise produces the same pin location as turning 330° counter-clockwise. Though not shown in Figure 1, at some points certain moves may not be possible (e.g. moving forward if a chair blocks the child) or even considered. If a child decides after turning that they are at the right location, they may only consider pushing the pin (i.e. the decision space has only one element, 'push pin').

Let us now define the decision spaces, alternative decision sequences, and interpretations considered by a recognition strategy α for an input $d_k \in D^g$. The series of decision spaces encountered by a recognition strategy α may be represented as a list of subsets of the game moves, T^{m^g}:

$$\alpha_{spaces}(\alpha, m^g, d_k \in D^g) = \{\{T_1^\alpha\} \subseteq T^{m^g}, .., \{T_{|s|}^\alpha\} \subseteq T^{m^g}\} \tag{9}$$

where $|s|$ is the length of the operation sequence produced by α. The complete set of alternative operation sequences considered for a given series of decision spaces $A = \alpha_{spaces}(\alpha, m^g, d_k \in D^g)$ and the related operation sequence s is then given by:

$$S(A, s) = \{\emptyset\} \cup \bigcup_{i=1}^{|s|} \bigcup_{a \in A_i} cat(s_{[1,i-1]}, a) \quad \subseteq L^{m^g} \tag{10}$$

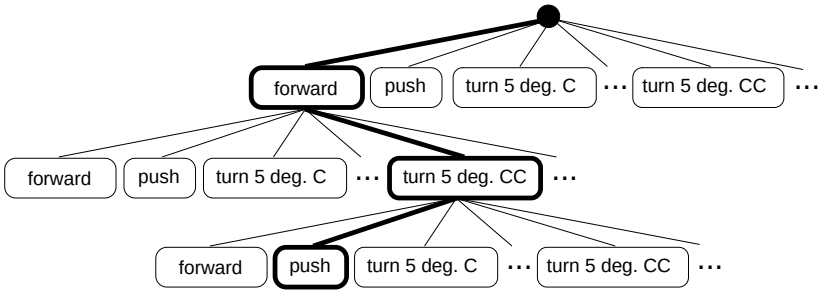

Fig. 1. Decision tree for one child's turn in 'pin the tail on the donkey.' The sequence of moves (*operation sequence*) shown is $s = (forward, turn\ 5\ deg.\ CC, push)$, representing the child stepping forward, turning five degrees counter-clockwise, and then pushing their tail-pin into a wall. '...' is used to represent turning clockwise and counter-clockwise in increments of five degrees (from 10° to 355°)

where *cat* appends elements of a decision space ($a \in A_i$) to subsequences of s from lengths 0 ($s_{[1,0]}$) to $|s|-1$, and $\{\emptyset\}$ is the empty sequence. S also describes the exhaustive set of paths from the root of the corresponding decision tree.

The complete set of points in the game's interpretation space (I^{m^g}) considered by recognition strategy α for input d_k is then given by:

$$C^{\alpha}_{d_k}(A, s) = \bigcup_{y \in S(A,s)} apply^{m^g}(y) \subseteq I^{m^g} \tag{11}$$

Equations (9), (10), and (11) allow us to analyze and compare recognition strategies quantitatively using decision spaces, operation sequences, and interpretations considered. For example, within a recognition game we can determine if a goal interpretation was considered by a strategy, and if so, in which decision space(s) (equivalently, at which nodes in the decision tree).

5 Example: A Table Cell Detection Game

We have previously carried out an informal recognition game for table cell detection [12]. We will now formalize the game, using our imitation game. In the game we compared the detection of table cells within lists of words and lines in segmented tables. We implemented two table recognition algorithms from the literature [13, 14] using the Recognition Strategy Language (RSL [12]). RSL formalizes decision making and captures operation sequences that transform interpretations represented as attributed graphs. One of the authors selected five challenging tables from the UW-I technical document corpus [15], and then produced a single set of table cells for each table. Both player algorithms return single interpretations. The distance from the algorithms' cell sets to the author's cell sets were measured using the harmonic mean of cell recall and precision (with a higher mean representing a smaller distance in the interpretation space).

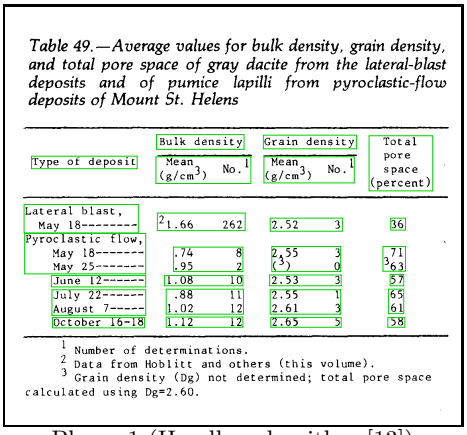

Table 49.—Average values for bulk density, grain density, and total pore space of gray dacite from the lateral-blast deposits and of pumice lapilli from pyroclastic-flow deposits of Mount St. Helens

Type of deposit	Bulk density Mean (g/cm³)	Bulk density No.[1]	Grain density Mean (g/cm³)	Grain density No.[1]	Total pore space (percent)
Lateral blast, May 18----	[2]1.66	262	2.52	3	36
Pyroclastic flow,					
May 18----	.74	8	2.55	3	71
May 25----	.95	2	([3])	0	[3]63
June 12----	1.08	10	2.53	3	57
July 22----	.88	11	2.55	1	65
August 7----	1.02	12	2.61	3	61
October 16-18	1.12	12	2.65	5	58

[1] Number of determinations.
[2] Data from Hoblitt and others (this volume).
[3] Grain density (Dg) not determined; total pore space calculated using Dg=2.60.

Player 1 (Handley algorithm [13])

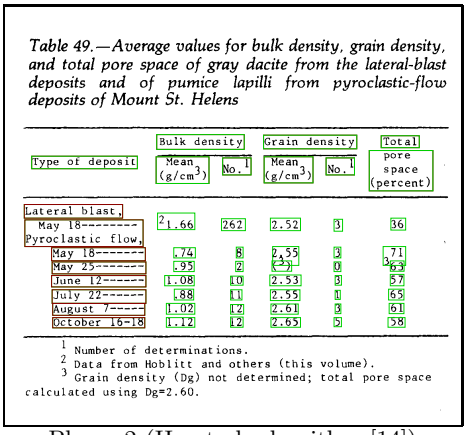

Table 49.—Average values for bulk density, grain density, and total pore space of gray dacite from the lateral-blast deposits and of pumice lapilli from pyroclastic-flow deposits of Mount St. Helens

Type of deposit	Bulk density Mean (g/cm³)	Bulk density No.[1]	Grain density Mean (g/cm³)	Grain density No.[1]	Total pore space (percent)
Lateral blast, May 18----	[2]1.66	262	2.52	3	36
Pyroclastic flow,					
May 18----	.74	8	2.55	3	71
May 25----	.95	2	([3])	0	[3]63
June 12----	1.08	10	2.53	3	57
July 22----	.88	11	2.55	1	65
August 7----	1.02	12	2.61	3	61
October 16-18	1.12	12	2.65	5	58

[1] Number of determinations.
[2] Data from Hoblitt and others (this volume).
[3] Grain density (Dg) not determined; total pore space calculated using Dg=2.60.

Player 2 (Hu et al. algorithm [14])

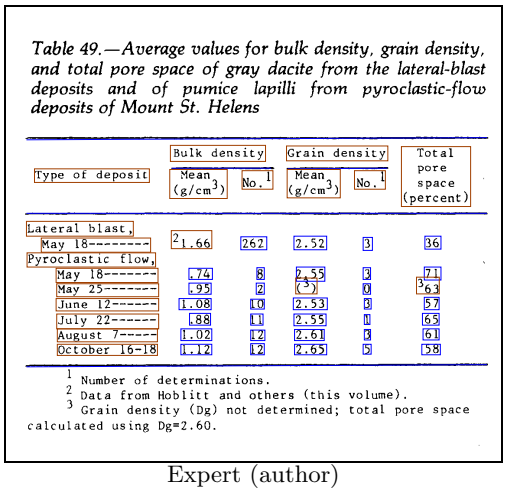

Table 49.—Average values for bulk density, grain density, and total pore space of gray dacite from the lateral-blast deposits and of pumice lapilli from pyroclastic-flow deposits of Mount St. Helens

Type of deposit	Bulk density Mean (g/cm³)	Bulk density No.[1]	Grain density Mean (g/cm³)	Grain density No.[1]	Total pore space (percent)
Lateral blast, May 18----	[2]1.66	262	2.52	3	36
Pyroclastic flow,					
May 18----	.74	8	2.55	3	71
May 25----	.95	2	([3])	0	[3]63
June 12----	1.08	10	2.53	3	57
July 22----	.88	11	2.55	1	65
August 7----	1.02	12	2.61	3	61
October 16-18	1.12	12	2.65	5	58

[1] Number of determinations.
[2] Data from Hoblitt and others (this volume).
[3] Grain density (Dg) not determined; total pore space calculated using Dg=2.60.

Expert (author)

Fig. 2. One round of a table cell detection game

The game had a 'best three out of five' outcome, where the algorithm with the highest harmonic mean would 'win' for each round (table). One round from the game is shown in Figure 2, for a table taken from page a038 of the UW-I corpus. For the round shown, Player 2 wins because both recall and precision are higher than for Player 1.

Let us now define the interpretation model m^c for our cell detection game. The problem domain D^{m^c} included the marked tables in the UW-I technical document corpus. The language of m^c was defined with $i_0^{m^c}$ equal to the empty set of cells, T^{m^c} containing basic operations of RSL that modify cell hypotheses (e.g. **accept**, **classify**, **relate**), and Γ^{m^c} defined by legal RSL operation sequences. The function $apply^{m^c}$ was defined by the interpretive layer of the RSL core that generates interpretations from operation sequences. The distance

metric δ^{m^c} was 1 - the harmonic mean of cell recall and precision, and $time^{m^c}_{max}$ was roughly 30 minutes, for both the author 'expert' and the algorithms.

The cell detection game parameters g included $m^g = m^c$, with $time^g_e = time^g_p = time^{m^c}_{max}$. Sampling was defined by μ^g (select 'challenging' tables), with sample size $n = 5$. Players were ranked using a 'best three out of five' protocol (ϕ^g). For the expert protocol $E^g = (A^e, \beta)$ the author acted as the sole 'expert', providing 1 interpretation per table ($A^e = \{author\}, \beta =$'choose single best interpretation'). The choice of β was significant, because often the author considered multiple interpretations. Finally, the players P^g were the two table recognition algorithms. One of the algorithms won; complete details may be found elsewhere [12].

The two player algorithms used different models of table structure. To address this, we isolated operations where cell hypotheses may have been affected. In terms of our imitation game, this might have been achieved using a more restrictive model language $\Gamma^{m^c}_r \subset \Gamma^{m^c}$ that included only sequences of RSL operations affecting cell hypotheses. The generated operation sequences could then have been filtered to include only these operations (a well-formed version of such an approach requires further investigation). For both algorithms, we were able to plot cell recall and precision against operation sequence positions affecting cell hypotheses, and observe new metrics describing the recall and precision of the *entire* set of cell hypotheses. The new metrics rely on the interpretation history (see equation (5)), and are called *historical recall* and *historical precision*. Using interpretation histories also simplified error analysis, particularly for locating which operations introduce errors [12].

In the future, we wish to modify RSL to automatically collect the decision space associated with each operation, given an input. We could then use the properties defined in the previous section to further compare recognition strategies. The game model might also provide the basis for a formal semantics of the RSL language.

6 Conclusion

The proposed imitation game defines the variables for a recognition task and the evaluation of proposed solutions explicitly, using an interpretation model and game parameters. This allows recognition tasks to be compared quantitatively in terms of the game variables (e.g. the relative sizes of interpretation spaces), and stipulates the terms of evaluation within the problem definition. Additionally, within the game expert and player moves are transparent, and may be compared using the decision trees they produce for inputs. In the future we are interested in defining general classes of strategies for specific games, as for example Cesa-Bianchi and Lugosi [8] have done, and begin considering *optimal* strategies for recognition games (similar to what Pau has done for Bayesian classification [10]).

For recognition tasks, we commonly talk about *ground-truth* as the set of correct interpretations for a problem domain. However, in practice ground-truth is comprised of interpretations for a sample of a problem domain, produced

by experts whose opinions may vary and even conflict [16]. We propose that this type of ground-truth is more accurately understood as expert opinion, as suggested in this paper.

Acknowledgments

We wish to thank Dr. Ching Y. Suen and CENPARMI for providing the resources to write this paper. We also wish to thank Burton Ma, Jeremy Bradbury, Chris McAloney, Dan Ghica, and the anonymous reviewers for their helpful comments. This research was funded by the Natural Sciences and Engineering Research Council of Canada.

References

1. Kanatani, K.: Uncertainty modeling and model selection for geometric inference. IEEE Trans. Pattern Analysis and Machine Intelligence **26** (2004) 1307–1319
2. Oommen, B., Rueda, L.: A formal analysis of why heuristic functions work. Artifical Intelligence **164** (2005) 1–22
3. Turing, A.: Computing machinery and intelligence. Mind (**59**) 433–460
4. Colman, A.: Game theory and experimental games: The study of strategic interaction. Pergamon Press, Oxford, England (1982)
5. Morris, P.: Introduction to Game Theory. Springer-Verlag, New York (1994)
6. Bozma, H., Duncan, J.: A game theoretic approach to integration of modules. IEEE Trans. Pattern Analysis and Machine Intelligence **16** (1994) 1074–1086
7. Chakraborty, A., Duncan, J.: Game-theoretic integration for image segmentation. IEEE Trans. Pattern Analysis and Machine Intelligence **21** (1999) 12–30
8. Cesa-Bianchi, N., Lugosi, G.: Potential-based algorithms in on-line prediction and game theory. Machine Learning **51** (2003) 239–261
9. Pau, L.: An adaptive signal classification procedure. Application to aircraft condition monitoring. Pattern Recognition (**9**) 121–130
10. Pau, L.: Game theoretical signal classification: Application to imperfect or non-cooperative learning. Geoexploration **23** (1984) 161–170
11. Zanibbi, R., Blostein, D., Cordy, J.: A survey of table recognition: Models, observations, transformations, and inferences. Int'l J. Document Analysis and Recognition **7** (2004) 1–16
12. Zanibbi, R.: A Language for Specifying and Comparing Table Recognition Strategies. PhD thesis, Queen's University, Kingston, Canada (2004)
13. Handley, J.: Table analysis for multi-line cell identification. In: Proc. Document Recognition and Retrieval VIII (IS&T/SPIE Electronic Imaging). Volume 4307. (2001) 34–43
14. Hu, J., Kashi, R., Lopresti, D., Wilfong, G.: Table structure recognition and its evaluation. In: Proc. Document Recognition and Retrieval VIII (IS&T/SPIE Electronic Imaging). Volume 4307. (2001) 44–55
15. Phillips, I., Chen, S., Haralick, R.: CD-ROM document database standard. In: Proc. Second Int'l Conf. Document Analysis and Recognition, Tsukuba Science City, Japan (1993) 478–483
16. Lopresti, D., Nagy, G.: Issues in ground-truthing graphic documents. In Blostein, D., Kwon, Y.B., eds.: Graphics Recognition: Algorithms and Applications. Volume 2390 of LNCS. Springer, Berlin (2002) 46–66